International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 07 | July - 2024 SJIF Rating: 8.448 ISSN: 2582-3930

DATA DUPLICATION USING FILE CHECKSUM

Akshatha[1], Dr.Shankaragowda B B[2]

[1]Student Department of MCA, BIET, Davanagere
[2]Head of the department,Depertment of MCA ,BIET, Davanagere

ABSTRACT

In modern computing systems, the collection of duplicate files causes problems ranging from wasted storage space to poor system performance. To address these concerns, a sophisticated program called the Duplicate File Searcher and Remover has been created. Using modern data deduplication methods, the program effectively detects duplicate files by comparing checksum data stored in a centralized database. Users can relocate duplicates for further examination or completely eliminate them, depending on their needs. Key goals include maximizing storage utilization, increasing data efficiency, and streamlining file management processes. The program caters to both administrators, who benefit from sophisticated file management features, and regular users, who enjoy smooth file uploading and management capabilities. By removing superfluous files, the utility improves system performance, speeds up indexing procedures, and decreases backup size and time. Overall, the Duplicate File Searcher and Remover is an important tool for improving data management, maximizing system efficiency, and increasing user productivity in digital contexts.

Keyword: Cloud computing, data storage, file checksum algorithms.

I. INTRODUCTION

In today's digital landscape, the growth of duplicate files presents substantial obstacles to effective file management and system performance. These redundant files, which accumulate over time, not only take up valuable storage space, but also hinder system responsiveness and complicate data structure. To address this issue, a sophisticated program has been created: the Duplicate File Searcher and Remover. This program uses complex data deduplication methods to quickly discover duplicate files by comparing checksum data from a central database. When duplicates are detected, the application allows users to either relocate them for further inspection or remove them completely, depending on individual preferences and organizational needs. The primary goals of this utility are to reclaim storage space, optimize resource utilization, and streamline file management processes. Users should expect improved data efficiency, faster indexing procedures, and shorter backup and storage times if unneeded duplicates are removed. below figure1 describe the architecture of the system.

The tool meets a wide range of operational requirements by incorporating separate modules for administrators and users. Administrators benefit from complete file management functions such as addition, modification, and deletion, which provide strong control over the system's content. Regular users, on the other hand, can easily upload files, explore their contents, and manage their uploads, resulting in a more user-friendly experience.

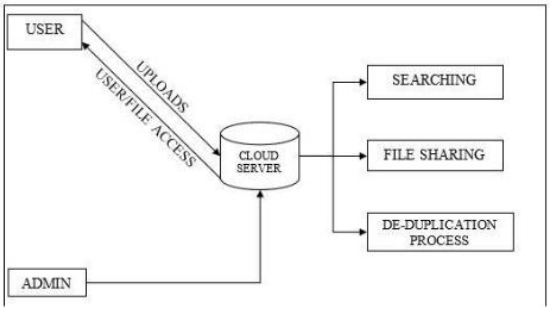

Fig1: Overall architecture of the system

In basic terms, the Duplicate File Searcher and Remover seeks to improve data management and system efficiency while also providing users with efficient, responsive working environments. This application, which effectively manages duplicate files, marks a critical step in optimizing digital workflows and increasing operational efficiency.

II.LITERATURE SURVEY

In "Di Pietro, Roberto, and Alessandro Sorniotti," the author proposes Proof of Ownership (POW) to solve security concerns associated with deduplication. POW are intended to allow the server to Determine whether a client has a file or not[1]. According to "Atishkathpal Matthew John Anf Gauravmakkar" as described by [2], data duplication reduction involves removing duplicate data from storage devices to reduce memory consumption. The system's underperformance stems from poor hardware management and user-friendliness, despite solid ideals. As corporations manage massive amounts of data, managing it becomes a significant task. Managing the deduplication of this data. To improve de-duplication scalability, previous efforts have focused on multi-threading [3], caching [4], and file segmentation. Implementing de-duplication requires extra controller resources, which may negatively impact I/O performance. Additionally, attempts have been made to optimize the algorithms in order to achieve better elimination of duplication effectiveness and output but none of these studies [5, 6] appear to have a both scalable architecture and have the same problems with the storage controller's excessive resource usage. For quicker results, attempts have also been made to do higher level de-duplication 0(file, object, or variable sized blocks) [7] as opposed to block level; nevertheless, they encounter scalability problems when dealing with billions of objects. Postprocess de-duplication processes are ideally scheduled during the "sleep" times in data center, during which the application load is expected to be much lower. However, the this kinds of presumptions are no longer true, with data centers servicing data globally. With an emphasis on non-binary alphabets, this study attempts to create a block-level deduplication method appropriate for files including edit errors in comparison to original versions. By reducing symbol comparisons in comparison to hash-based and brute-force techniques, the algorithm seeks to achieve significant deduplication ratios. To prove the algorithm's efficacy and efficiency in real-world situations, the paper offers both theoretical analysis and actual data[8]. In order to reduce errors brought about by duplicate patient data in the virtual repositories of medical organizations, this paper presents the Checksum Method for Data Duplication (CMDD), a novel approach implemented on the Java Platform. The paper offers experimental findings that show how CMDD can effectively detect and handle data duplication, boosting data accuracy and raising the standard of care in medical settings[9]. The objective of this research is to create and apply a technique for removing duplicate data by leveraging AWS (Amazon Web Services) resources. The goal is to efficiently find and remove duplicate data instances from huge datasets by utilizing AWS's cloud infrastructure and technologies. By incorporating cutting-edge AWS services designed specifically for data deduplication, the project aims to increase overall system performance, optimize storage efficiency, and improve data quality[10].

III.METHODOLOGY

The methods used to create the Duplicate File Searcher and Remover application sought to provide a reliable solution for properly managing duplicate files. The tool was created with requirements gathering, systematic design, rigorous testing, and user feedback in mind to improve data organization, optimize system efficiency, and increase user productivity in digital

contexts. The Duplicate File Searcher and Remover program was developed using a structured method to provide robust functionality and usability.

A. Requirements Gathering:

Initially, thorough needs were acquired through interactions with stakeholders, such as system administrators and end users. This phase outlined important capabilities such as file uploading, duplication detection methods, database management, and user interface design.

B. Design and Architecture:

A detailed design and architecture were developed in response to the needs received. The system architecture was created to facilitate scalability, with a modular approach that distinguished between administrative and user operations. The design stressed the use of efficient data deduplication methods to provide accurate and quick duplicate identification.

C. Implementation:

The tool was built with a variety of programming languages and technologies designed for efficient file processing and database management. Checksum calculation and comparison algorithms were incorporated to ensure that duplicate files were correctly identified. The implementation phase also focused on creating a user-friendly interface that allows users to handle their files in an easy manner.

D. Testing and Validation:

Rigorous testing processes were used to ensure the tool's functionality and performance. Unit tests confirmed the accuracy of individual modules, whereas integration tests ensured that diverse components interacted seamlessly. Performance testing assessed the tool's ability to handle large numbers of files and detect duplicates within reasonable time constraints.

E. Deployment and User Training:

Following satisfactory testing, the tool was deployed in a controlled environment. User training sessions were held to educate administrators and end-users with the tool's functionality, such as file management, duplicate handling, and system maintenance.

F. Documentation:

Comprehensive documentation was created, including installation instructions, user guides, and administration manuals. This documentation made it easier to deploy and adapt the system, as well as troubleshoot and maintain it.

3.1 Dataset Used

In the field of "Data Deduplication Using File Checksum," a number of important datasets are essential to the development and assessment of successful deduplication algorithms. Firstly, a realistic dataset for detecting duplicate instances is provided via a collection of sample files that span many types, including text documents, photos, and multimedia. These files model common data scenarios in which system inefficiencies or user error may cause duplication. A checksum database, which has computed checksum values for every file in the sample dataset, is also necessary. During the deduplication process, this database serves as a reference, making it possible to quickly compare and identify duplicate files based on their checksums. The algorithm's capacity to identify and handle duplication is further validated using test data, which includes purposefully duplicated files.

3.2 Data Preprocessing

To achieve precise and effective duplicate detection, there are numerous crucial phases in the data preprocessing for file checksum-based data deduplication. First, a variety of sample files are gathered to mimic common data sources. Preprocessing is applied to every file in order to retrieve metadata and generate a unique checksum depending on the content, which functions as a fingerprint for quick comparison. A centralized database contains these checksums so that they may be easily accessed during the deduplication process. To improve algorithm accuracy, preprocessing can also address encoding variances, standardize formats, and normalize metadata. By simplifying duplicate identification, these methods improve data management procedures and storage effectiveness.

3.3 Algorithm Used

Checksum Algorithms

A cryptographic known as Secure Hash Algorithm 256, or SHA-256 takes an input file and outputs a 256-bit (32-byte) hash result. In order to generate a unique

hash value, it processes 512-bit blocks of incoming data and performs a number of mathematical operations. By acting as a digital fingerprint for the input data, this hash value guarantees that even a small alteration to the input will produce a noticeably different hash output.

SHA-256 is used in the context of data deduplication to calculate file checksums or hash values. Throughout the deduplication process, these checksums are quickly compared using data that is kept in a database. When a checksum from a recently uploaded file matches one that already exists in the database, It suggests that there is more than one file. By efficiently managing redundant files and identifying them, deduplication systems maximize storage capacity and enhance the effectiveness of data management as a whole. Because of its deterministic and collision-resistant characteristics, SHA-256 is a key technique in contemporary data deduplication solutions, ensuring consistent and reliable identification of duplicate files across enormous datasets.

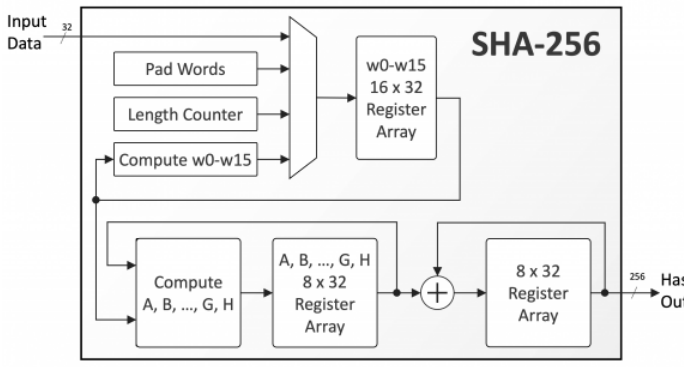

Figure2 : architecture diagram SHA 256

3.4 Techniques Used

Several strategies are used in data deduplication to effectively manage and minimize redundant data. Content-based chunking uses techniques such as Rabin fingerprinting to discover similar chunks and separates files into pieces based on content rather than predetermined sizes. By minimizing duplicate data, delta encoding optimizes storage by recording only the changes (delta) between file versions. Compression techniques use compressed data to make comparisons and reduce file sizes by removing repeating patterns. With hash-based deduplication, files are assigned unique identifiers (hashes), which are then compared to identify duplicates. By identifying redundant patterns in files, machine learning techniques improve the accuracy of deduplication. These methods, which are customized for different applications, are integrated by dedicated data deduplication systems to increase data efficiency and simplify storage management.

IV. RESULT AND DISCUSSION

Data deduplication with file checksums improves storage efficiency by finding and removing duplicate data based on unique file identities created by techniques such as MD5, SHA-1, or SHA-256. This strategy minimizes storage requirements, resulting in cost reductions in hardware and operational expenses, which are especially helpful in large-scale storage systems such as data centers. It minimizes bandwidth utilization by sending only unique data chunks over networks, resulting in faster data transfer speeds.

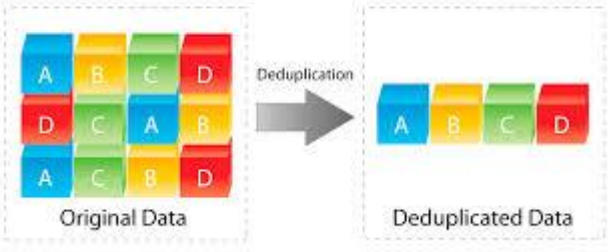

figure 3: example of data duplication

Checksum-based deduplication accelerates backup procedures and allows for faster data recovery by saving only unique data blocks. Despite issues such as potential hash collisions and computational overhead, checksum-based deduplication is nevertheless critical to maintaining data integrity and boosting overall system efficiency in modern IT infrastructures.

V. CONCLUSION

According to the features and benefits listed, the Duplicate File Searcher and Remover is an important solution in modern computer environments. By efficiently applying data deduplication methods via checksum comparisons kept in a centralized database, the application addresses significant concerns such as wasted storage space and slow system performance caused by duplicate files. Its user-friendly features cater to both administrators and normal users, allowing for effective file management by relocating or eliminating duplicate files. Key objectives such as increasing storage usage, improving data efficiency, and optimizing file operations highlight its importance in boosting overall system performance. Finally, the utility greatly improves data management, optimizes system efficiency, and boosts user productivity in digital environments.

VI.REFERENCES

[1]. Di Pietro, Roberto and Alessandro Sorniotti, "Proof of ownership for de-duplication systems: A secure, scalable, and efficient solution", Computer Communications, 15 May 2016.

[2]. Atishkathpal, Matthew John and Gauravmakkar, "Distributed Duplicate Detection in Post-Process Data De-duplication", Conference: HiPC , 2011.

[3]. Petros Efstathopoulos and Fanglu Guo. Rethinking Deduplication Scalability. HotStorage 2010.

[4]. ZHU, B., LI, K., AND PATTERSON, H. Avoiding the disk bottleneck in the Data Domain deduplication file system. In File and Storage Technology Conference (2008).

[5]. Sean Quinlan and Sean Dorward, “Venti: A new approach to archival storage,” in FAST ’02: Proceedings of the Conference on File and Storage Technologies, Berkeley, CA, USA, 2002, pp. 89–101, USENIX Association.

[6] OpenDedup, “A userspace deduplication file system (SDFS),” March 2010, http://code.google.com/p/opendedup/.

[7]. Shai Halevi, Danny Harnik, Benny Pinkas, and Alexandra Shulman-Peleg. Proofs of Ownership in Remote Storage Systems. CCS 2011.

[8]. L. Conde-Canencia, T. Condie and L. Dolecek, "Data Deduplication with Edit Errors," 2018 IEEE Global Communications Conference (GLOBECOM), Abu Dhabi, United Arab Emirates, 2018, pp. 1-6, doi: 10.1109/GLOCOM.2018.8647415.

[9]. A. Razaque, M. Almiani, B. Alotaibi, M. Alotaibi and M. Alweshah, "Novel File-Checksum Method for Data Duplication Removal of Patients," 2021 Eighth International Conference on Software Defined Systems (SDS), Gandia, Spain, 2021, pp. 1-5, doi: 10.1109/SDS54264.2021.9732163.

[10]. S. Singhal, P. Gupta, S. Singh, P. Agarwal and K. Kumar, "Data Duplication Removal Technology Using AWS Services," 2021 5th International Conference on Information Systems and Computer Networks (ISCON), Mathura, India, 2021, pp. 1-4, doi: 10.1109/ISCON52037.2021.9702380.